Muri

Il problema

In un certo Paese, sono stati costruiti grandi muri di delimitazione in modo che ogni muro connetta esattamente due città. I muri non si incrociano fra loro. Quindi, il Paese risulta diviso in regioni tali che per spostarsi da una regione all'altra è necessario passare attraverso una città o un muro. Per ogni coppia di città A e B esiste al più un muro con un'estremità in A e una in B; inoltre, è sempre possibile andare da una città a un'altra camminando sempre dentro città o lungo muri. Il formato dell'input impone ulteriori restrizioni.

C'è un club i cui membri vivono in alcune delle città. In ogni città vive al massimo un membro. A volte, i membri vogliono incontrarsi in una delle regioni, fuori da ogni città. I membri viaggiano in bicicletta, e non vogliono entrare in nessuna città a causa del traffico. Inoltre, vogliono attraversare il minor numero di muri possibile, perché attraversare un muro è molto complicato. Per arrivare alla regione d'incontro, ogni membro deve attraversare un certo numero (eventualmente 0) di muri. I membri vogliono trovare una regione tale che la somma di questi numeri (in breve, la *somma degli attraversamenti*) sia minima.

Le città sono etichettate con interi da 1 a N, dove N è il numero di città. In figura 1, i nodi etichettati rappresentano le città e le linee che connettono i nodi rappresentano i muri. Supponiamo che ci siano tre membri, che vivono nelle città 3, 6 e 9. In tal caso, una regione ottima e le rispettive strade per i membri sono mostrate in figura 2. La somma degli attraversamenti è 2: il membro che vive nella città 9 deve attraversare il muro fra le città 2 e 4, mentre il membro che vive nella città 6 deve attraversare il muro fra le città 4 e 7.

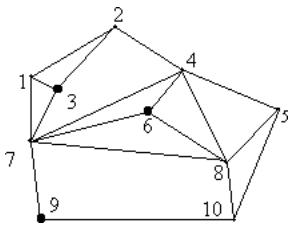

Figura 1

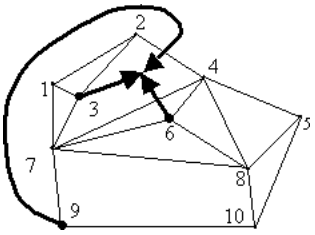

Figura 2

Dovete scrivere un programma che, data la descrizione delle città, delle regioni e delle città in cui vivono i membri del club, calcoli una regione ottima e la minima somma di attraversamenti.

Dati in input

Il nome del file di input è `WALLS.IN`. La prima riga contiene un intero: il numero di regioni M, $2 \leq M \leq 200$. La seconda riga contiene un intero: il numero di città N, $3 \leq N \leq 250$. La terza riga contiene un intero: il numero di membri L, $1 \leq L \leq 30$, $L \leq N$. La quarta riga contiene L interi distinti in ordine crescente: le etichette delle città dove vivono i membri.

Dopo questi dati, il file contiene $2M$ righe, sicché ci sono due righe per ogni regione: le prime due righe delle $2M$ descrivono la prima regione, le seguenti due la seconda e così via. In ciascuna coppia, la prima riga indica il numero di città I sul bordo di quella regione. La seconda riga della coppia contiene le etichette di queste I città in un ordine in cui possono essere attraversate facendo un giro in senso orario lungo il bordo della regione, con la seguente eccezione: l'ultima regione è la regione *esterna*, che circonda tutta la città, e per essa l'ordine delle etichette corrisponde a un giro in senso antiorario. L'ordine delle regioni fornisce loro implicitamente un'etichettatura costituita da un numero intero: la prima regione ha etichetta 1, la seconda ha etichetta 2 e così via. Notate che l'input include tutte le regioni formate dalle città e dai muri, inclusa la regione esterna.

Dati in output

Il nome del file di output è `WALLS.OUT`. La prima riga contiene un intero: la minima somma di attraversamenti. La seconda riga contiene un intero: l'etichetta di una regione ottima. Ci potrebbero essere molte regioni diverse che sono soluzioni del problema, ma il vostro programma deve emetterne in output una sola.

Esempio di input e output

WALLS.IN	WALLS.OUT
10 10 3 3 6 9 3 1 2 3 3 1 3 7 4 2 4 7 3 3 4 6 7 3 4 8 6 3 6 8 7 3 4 5 8 4 7 8 10 9 3 5 10 8 7 7 9 10 5 4 2 1	2 3